Popular Computing

1

Vol 1 No 1 April 1973

BOX 272
CALABASAS, CALIFORNIA 91302

The 3X + 1 Problem

The 3X + 1 problem was devised in 1967 by Prof. Richard Andree of the University of Oklahoma, to demonstrate the concepts of recursion and algorithms to a class of high school students. The problem is this: for a positive integer, N, let X equal N and proceed as follows:

> Replace X by X/2 if X is even.
> Replace X by 3X+1 if X is odd.
> Stop when X equals 1.
> Call the number of terms so generated A, counting the original number.

Thus, for N = 9, the generated sequence is 9, 28, 14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1, and A = 20.

Notice that in performing this calculation, we have certified the convergence of many other values of N, and also determined their A values.

The first natural question that arises is, will the process converge for all values of N? This seems to be difficult to prove. Convergence has been established by computation for all values of N up to 200,000,000, utilizing the following shortcuts:

1. Only odd values of N need be tested, if the computation proceeds upwards in the N values, since for any even N the half value will then already have been tested.

2. It is not necessary to proceed to X = 1 to prove convergence, but only to a value of X less than N.

3. Numbers of the form 4K+1 will follow this sequence:

4K+1
12K+4
6K+2
3K+1

and hence will always converge, provided that values of N are tested in increasing order. Thus, only odd values of N of the form 4K+3 need be tested. The table of Figure 1 shows the slow growth of values of A.

The A values seem to cluster in a peculiar way. For example, these values of N all have an A of 39:

610, 611, 612, 613, 614,
628, 629, 630, 631, 632.

For any limited range of N, there will be comparatively few values of A. Thus, in the range of N from 90,000 through 94,999, over half of the A values are these: 85, 116, 134, 147, and 178.

An N can be found for any given A. Thus, for $A = k$, use $N = 2^{k-1}$. If values of N are restricted to being odd, then not every A can be produced, and empirical studies indicate that relatively few A values exist.

POPULAR COMPUTING is published monthly at Box 272, Calabasas, California 91301. Subscription rate in the United States is $15 per year, or $12 if remittance accompanies the order. For all foreign subscriptions, add $4.50 per year. Multiple subscriptions to the same address, add $5 each; thus, 3 copies per month is $25 per year, U.S. delivery. Back issues $1.50 each. Subscriptions may begin with any issue. Subscriptions for qualified undergraduate students half price.

A	N	A	N
1	1	282	23529
8	3	308	26623
17	7	311	34239
20	9	324	35655
21	19	340	52527
24	25	351	77031
112	27	354	106239
113	55	375	142587
116	73	383	156159
119	97	386	216367
122	129	443	230631
125	171	449	410011
128	231	470	511935
131	313	509	626331
144	327	525	837799
145	649	528	1117065
171	703	531	1501353
179	871	557	1723519
182	1161	560	2298025
183	2223	563	3064033
209	2463	584	3542887
217	2919	597	3732423
238	3711	613	5649499
262	6171	665	6649279
268	10971	686	8400511
276	13255	689	11200681
279	17647	692	14934241
		705	15733191

Figure 1. Results from the 3X + 1 problem. The table shows the first appearance of successively larger values of A, with the corresponding N. Only odd values of N were being examined.

What remains to be done with the 3X+1 problem? Values of N greater than 200,000,000 could be tested for convergence (although this path is losing its charm). One new computer approach would be the following. Clear to zero a block of bits each of which represents an odd integer. Now, in testing an odd N for convergence, store a 1 at the bit position corresponding to any odd number greater than N that is met along the way. Thus, in testing N = 9, the bit positions corresponding to values of 11, 17, and 13 would be set to 1. The remaining zeros in the block of bits would then point to the next value of N that needs to be tested. The shortcuts listed earlier can still be applied.

It might be concluded that the process will probably always converge, and the proper avenue of attack on the convergence problem will be analytic. As a computer problem, work needs to be done on the distribution of the A values.

--JJJ

The Wells/Ulam Conjecture

Writing in the September, 1964 Scientific American, Stanislaw Ulam reported on a conjecture made by Mark Wells, to the effect that multiples of 3, expressed in binary notation, should most often have an even number of 1-bits. This is stated as a theorem: "Among all the integers divisible by three from 1 to 2^N, those that have an even number of 1's always predominate, and the difference between their number and the number of those with an odd number of 1's can be computed exactly: it is $3^{(N-1)/2}$."

The table shows actual counts up to 2^{16}, plus two more observations that are not at powers of 2. The value given by the formula is rounded to the nearest integer. There is also shown the ratio of odds to evens, which seems to tend toward 1.000 as N increases. The repetition in the ratio indicated by the brackets on the far right is probably significant.

N	2^N	Number of multiples of 3 having an even number of 1-bits	Number of multiples of 3 having an odd number of 1-bits	Difference	$3^{(N-1)/2}$	Ratio of odds to evens
1	2	0	0	0	1	.000
2	4	1	0	1	2	.000
3	8	2	0	2	3	.000
4	16	5	0	5	5	.000
5	32	9	1	8	9	.111
6	64	19	2	17	16	.105
7	128	34	8	26	27	.235
8	256	69	16	53	47	.232
9	512	125	45	80	81	.360
10	1024	251	90	161	140	.359
11	2048	462	220	242	243	.476
12	4096	925	440	485	421	.476
13	8192	1729	1001	728	729	.579
14	16384	3459	2002	1457	1263	.579
15	32768	6554	4368	2186	2187	.666
16	65536	13109	8736	4373	3788	.666
	500000	92463	74203	18260		.802
	1000000	184900	148433	36467		.802
	1500000	270475	229525	40950		.849

Hewlett-Packard HP-35

The HP-35 is unquestionably the leader in the pocket calculator field; it is the machine against which all others must be compared. It is in a class by itself, both on price ($400), its precision (10 significant digits), and its functions (natural and common logarithms, exponential, power function, direct and inverse sine, cosine, and tangent). A 4-level stack storage allows elaborate cascading of operations and forces the use of Polish (Lukasiewicz) notation. Thus, for the operation A/B = Q, most calculators follow the sequence: enter A, press divide, enter B, press equals. On the HP-35, the sequence is: enter A, enter B, press divide. The machine also has, in addition to the 4-level stack, one additional storage word.

Generally, the claim for 10 significant digit precision holds good. Inverse operations such as

X, reciprocal, reciprocal = X

$X, e^X = Q, \ln Q = X$

return to the exact original value for a wide range on X.

All arithmetic is floating point in the range from 10^{-2} to 10^{10} and outside that range is in scientific notation, up to 9.999999999E99. Illegal operations, such as division by zero or the logarithm of a negative number cause the display to blink. The sequence of operations: enter 2, square root, square root, square root, square root, square root, square, square, square, square, square, equals 2.000000022 indicates the inherent accuracy of the algorithms that have been programmed into the machine. On the other hand, the sine function appears to apply an algorithm directly without reduction by multiples of two pi; thus, sine 30 = .5 but sine 750 = .4999999986. Similarly, sine 720 = 4E-09.

Early models of the HP-35 had an error in one of the LSI chips. In a mailing to registered owners, the company stated "In most cases, the error range in the answers is from one one-hundreth (sic) of one percent to a maximum of one percent." The first such error

listed in that mailing is:

arcsin .0002 = 5.729577893E-3 (HP-35)
arcsin .0002 = .01145916 (true)

which is a 50% error.

In any event, this set of bugs has been corrected on later serial numbers, and Hewlett-Packard has agreed to modify the early machines.

The HP-35 is a BEST BUY by any criterion. No user, after a few weeks of use, would be likely to want to go back to a simpler machine at any price. The machine represents a significant jump in calculator capability and, after a year of sales, has no competitor.

THE CALENDAR

The present calendar in use throughout most of the world began in 1582; it was adopted by Great Britain and the Colonies in 1752. The calendar repeats precisely every 400 years. The tables below show the statistics for any period of 400 successive years.

Table 1 shows the distribution of month types for the 4800 months. There are 44 months of 28 days that begin on Sunday; 399 months of 31 days that begin on Saturday; and so on.

Table 2 shows the distribution of days of the month. For example, the 13th of the month falls on Friday 688 times, which is more often than any other day of the week.

Table 3 shows the distribution of the 14 year types. Thus, 44 years are not leap years and begin on a Thursday. Of the 97 leap years, 15 will begin on a Friday.

Table 4 shows the years in the current century that are of each of the 14 possible types.

Table 1:

	S	M	T	W	T	F	S
28 day month	44	43	43	43	43	44	43
29 day month	13	15	13	15	13	14	14
30 day month	230	228	229	228	228	229	228
31 day month	401	398	402	399	401	400	399

Table 2:

688	684	687	685	685	687	684	1
684	688	684	687	685	685	687	2
687	684	688	684	687	685	685	3
685	687	684	688	684	687	685	4
685	685	687	684	688	684	687	5
687	685	685	687	684	688	684	6
684	687	685	685	687	684	688	7
688	684	687	685	685	687	684	8
684	688	684	687	685	685	687	9
687	684	688	684	687	685	685	10
685	687	684	688	684	687	685	11
685	685	687	684	688	684	687	12
687	685	685	687	684	688	684	13
684	687	685	685	687	684	688	14
688	684	687	685	685	687	684	15
684	688	684	687	685	685	687	16
687	684	688	684	687	685	685	17
685	687	684	688	684	687	685	18
685	685	687	684	688	684	687	19
687	685	685	687	684	688	684	20
684	687	685	685	687	684	688	21
688	684	687	685	685	687	684	22
684	688	684	687	685	685	687	23
687	684	688	684	687	685	685	24
685	687	684	688	684	687	685	25
685	685	687	684	688	684	687	26
687	685	685	687	684	688	684	27
684	687	685	685	687	684	688	28
644	641	644	642	642	643	641	29
627	631	626	631	627	629	629	30
400	399	401	398	402	399	401	31

Table 3:

Not leap year	43	43	44	43	44	43	43
Leap year	15	13	14	14	13	15	13

Table 4:

Normal years starting with Sunday

1905 1911 1922 1933 1939 1950 1961 1967 1978 1989
1995 2006

Normal years starting with Monday

1900 1906 1917 1923 1934 1945 1951 1962 1973 1979
1990 2001

Normal years starting with Tuesday

1901 1907 1918 1929 1935 1946 1957 1963 1974 1985
1991 2002

Normal years starting with Wednesday

1902 1913 1919 1930 1941 1947 1958 1969 1975 1986
1997 2003

Normal years starting with Thursday

1903 1914 1925 1931 1942 1953 1959 1970 1981 1987
1998 2009

Normal years starting with Friday

1909 1915 1926 1937 1943 1954 1965 1971 1982 1993
1999 2010

Normal years starting with Saturday

1910 1921 1927 1938 1949 1955 1966 1977 1983 1994
2005

Leap years starting with Sunday

1928 1956 1984 2012

Leap years starting with Monday

1912 1940 1968 1996 2024

Leap years starting with Tuesday

1924 1952 1980 2008

Leap years starting with Wednesday

1908 1936 1964 1992 2020

Leap years starting with Thursday

1920 1948 1976 2004

Leap years starting with Friday

1904 1932 1960 1988 2016

Leap years starting with Saturday

1916 1944 1972 2000

Book Review

Program Test Methods, edited by William C. Hetzel, Prentice-Hall, 1973, 311 pages plus a bibliography of 375 citations plus an Index of Concepts.

This is the Proceedings of a conference on Computer Program Test Methods, sponsored by the University of North Carolina and the ACM Special Interest Group on Programming Languages, and held at Chapel Hill June 21-23, 1972. It is the first attempt to bring together what is known about software validation.

Program testing--how to certify a program as performing the way it should--ought to be a fundamental concept in the learning of computing. As a basic concept, it should be an important topic in introductory texts. But an examination of any dozen textbooks taken at random will reveal how carefully the subject has been avoided. The majority of texts ignore the subject completely. Those that do mention it usually get it nicely confused with debugging and then compound the confusion in the student's mind by offering him fatuous or worthless guidelines for devising test procedures. It would not be farfetched to say that beginning students are systematically being taught how to produce garbage with computers.

To quote from the book's Preface:

> The growth of a testing discipline has been painfully slow, despite the acknowledged need for better quality software. Even the scope of what is or is not a testing activity is not well defined. Terminology in the field is unclear and literature that would establish some foundations has not been written. Testing as an activity needs and requires more attention.
>
> A sense of increasing urgency seems prevalent and much research is underway. In general, however, the methodology, techniques and theory of program testing are entirely inadequate. It is hoped that this book is a start toward the solutions needed.

The 23 papers in the book are a beginning, at least. Each of the eight sections begins with some pertinent quotations, one of which is worth repeating:

> In the space of one hundred and seventy-six years the Lower Mississippi has shortened itself two hundred and forty-two miles. That is an average of a trifle over one mile and a third per year. Therefore, any calm person, who is not blind or idiotic, can see that in the old Oolitic Silurian Period, just a million years ago next November, the Lower Mississippi River was upward of one million three hundred thousand miles long, and stuck out over the Gulf of Mexico like a fishing-rod. And by the same token any person can see that seven hundred and forty-two years from now the Lower Mississippi will be only a mile and three quarters long, and Cairo and New Orleans will have joined their streets together, and be plodding comfortably along under a single mayor and a mutual board of aldermen. There is something fascinating about science. One gets such wholesale returns of conjecture out of such a trifling investiment of fact.
>
> --Mark Twain (Life on the Mississippi)

Program Test Methods is certain to become one of the milestones in software literature. The various authors addressed themselves to the stated problem; they suggested feasible criteria for software certification; a methodology to apply to future software projects; and the boundaries of current capability in this most sensitive field.

--JJJ

SUBFACTORIALS

Subfactorials are defined by the recursion:

$$!N = N \cdot !(N-1) + (-1)^N$$

with subfactorial 1 defined as zero. In the following table, all values up to !28 are exact; for those that follow, the first 30 significant digits are given, rounded from the 31st digit, followed by the number of digits to the decimal point.

1	0		31	302501328894190910970370027530	004
2	1		32	968004252461410915105184088096	005
3	2		33	319441403312265601984710749072	007
4	9		34	108610077126170304674801654684	009
5	44		35	380135269941596066361805791395	010
6	265		40	300158458444475693321518926221	018
7	1854		50	111887196107824805046302580708	035
8	14833		60	306112008903007593241410795996	052
9	133496		70	440667025198059441726775008254	068
10	1334961		80	263289318631230725980923677587	089
11	14684570		90	546564358753040868507836153046	108
12	176214841		100	343327959841638047651959775268	128
13	2290792932				
14	32071101049				
15	481066515734				
16	7697064251745				
17	130850092279664				
18	2355301661033953				
19	44750731559645106				
20	895014631192902121				
21	18795307255050944540				
22	413496759611120779881				
23	9510425471055777937262				
24	228250211305338670494289				
25	5706255282633466762357224				
26	148362637348470135821287825				
27	4005791208408693667174771274				
28	112162153835443422680893595673				
29	325270246122785925774591427452	001			
30	975810738368357777323774282355	002			